DESIGN AND IMPLEMENTATION OF A MODULAR HUMAN-ROBOT INTERACTION FRAMEWORK

A Thesis

presented to

the Faculty of California Polytechnic State University,

San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science in Mechanical Engineering

by

Michael Juri

June 2021

COMMITTEE MEMBERSHIP

TITLE: Design and Implementation of a Modular Human-Robot Interaction Framework

AUTHOR: Michael Juri

DATE SUBMITTED: June 2021

COMMITTEE CHAIR: Eric Espinoza-Wade, Ph.D.
Professor of Mechanical Engineering

COMMITTEE MEMBER: Charlie T. Refvem
Lecturer of Mechanical Engineering

COMMITTEE MEMBER: William R. Murray, Ph.D.
Professor of Mechanical Engineering

ABSTRACT

Design and Implementation of a Modular Human-Robot Interaction Framework

Michael Juri

With the increasing longevity that accompanies advances in medical technology comes a host of other age-related disabilities. Among these are neuro-degenerative diseases such as Alzheimer's disease, Parkinson's disease, and stroke, which significantly reduce the motor and cognitive ability of affected individuals. As these diseases become more prevalent, there is a need for further research and innovation in the field of motor rehabilitation therapy to accommodate these individuals in a cost-effective manner. In recent years, the implementation of social agents has been proposed to alleviate the burden on in-home human caregivers. Socially assistive robotics (SAR) is a new subfield of research derived from human-robot interaction that aims to provide hands-off interventions for patients with an emphasis on social rather than physical interaction. As these SAR systems are very new within the medical field, there is no standardized approach to developing such systems for different populations and therapeutic outcomes. The primary aim of this project is to provide a standardized method for developing such systems by introducing a modular human-robot interaction software framework upon which future implementations can be built.

The framework is modular in nature, allowing for a variety of hardware and software additions and modifications, and is designed to provide a task-oriented training structure with augmented feedback given to the user in a closed-loop format. The framework utilizes the ROS (Robot Operating System) middleware suite which supports multiple hardware interfaces and runs primarily on Linux operating systems. These design requirements are validated through testing and analysis of two unique implementations of the framework: a keyboard input reaction task and a reaching-

to-grasp task. These implementations serve as example use cases for the framework and provide a template for future designs. This framework will provide a means to streamline the development of future SAR systems for research and rehabilitation therapy.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Lengthening lifespans have increasingly resulted in more people aging into and with disabilities [1]. The resulting increased burden on health care systems has motivated alternative approaches to the delivery of critical health services. One class of diseases for which there is an increasing gap between individuals in need and available services is neuro-degenerative diseases, including Alzheimer's disease, Parkinson's disease, and stroke. According to the American Heart Association, stroke is a leading cause of serious long-term disability in the United States, reducing mobility in more than half of stroke survivors age 65 and over [2]. Many individuals worldwide suffer from motor impairments due to stroke and other conditions, making daily life more challenging. To improve their quality of life and regain motor function, these individuals must undergo motor rehabilitation in the form of physical or occupational therapy. Ideally, this therapy is personalized for each individual to maximize the efficiency and effectiveness of their treatment. This optimization utilizes support and feedback that adapts to the patient's needs, providing an appropriate level of challenge to match the patient's physical and cognitive ability. Unfortunately, this specialized therapy can be extremely resource intensive, costing a large amount of time and money for in-home rehabilitation.

In recent years, the use of technology to alleviate caregiver burden has increased due to advances in the fields of mobile health, wearable sensing, and the Internet of Things (IoT). A more recent development for telehealth and in-home applications is socially-assistive robotic (SAR) systems. SAR systems are generally applied in domains where hands-off, social interaction may provide therapeutic benefits to the human user and

can be used to alleviate the cost of in-home therapy. For therapeutic interactions requiring long-term, supervised care such as motor or neurological rehabilitation, an SAR platform may be an ideal extension of care into a user's home setting. By utilizing software-driven feedback systems, SARs can facilitate current, evidence-based strategies for such rehabilitation. SAR systems, coupled with environmental or wearable sensors for detecting human behavior, can essentially be treated as a closed-loop feedback system in which the robot, through various forms of feedback, drives the individual towards a desired state. Utilizing this robot feedback system can eliminate the potential costs of having an in-home human clinician while providing the ability to specialize the patient's therapy on an individual basis by altering the robot's software. Additionally, the modality of robot feedback can be adapted to the specific communication needs of the user.

In spite of the potential merits of SAR systems, many unknown factors must be resolved in order to ensure their applicability in the health care domain. These unknown factors include, but are not limited to: how user performance and learning are influenced by the design of the robot system; the independent variables associated with therapy (e.g., time-on-task, practice intensity); the medium of feedback delivery; and the adaptability of robot responses to user behaviors. Investigation of these factors requires a system for which independent variables may be objectively adjusted.

The focus of this project is the design of a novel human-robot interaction framework that can be used as a testbed for motor-neurological rehabilitation experiments. The realization of this design requires definition of the need for ambient, telehealth systems and the underlying theoretical framework requirements of task-oriented training, augmented feedback, embodiment, and modular flexibility (Ch. 2). In the upcoming chapters, the specific design process of this system is discussed (Ch. 3) and its design requirements are validated by presenting multiple implementations of the system and

their use cases (Ch. 4). Finally, the results of this project are contextualized and the potential for future work and development of the framework is discussed (Ch. 5).

Through research, design, implementation, and testing, the novel human-robot interaction framework was successfully created and validated with respect to the design requirements. The final products of this project consist of two working implementations of the framework that demonstrate its features and use cases, providing examples of how future SAR systems may be constructed from the underlying framework. Testing and analysis of these two implementations proves that the framework is robust and easy to use. In addition, the framework has built-in support for many different software libraries and drivers for interfacing with various hardware. Finally, all software will be made available in a public repository at the completion of initial testing and dissemination.

This framework is designed to be built upon for future SAR systems to facilitate both SAR research and motor rehabilitation therapy. The novelty of this framework is its structure, which provides a modular template for future implementations that retain the work flow and feedback characteristics of the basic framework. By using the framework and simply adding functionality to it, the development of SAR systems should be far more streamlined as a large portion of the software development and design is inherent to the framework and thus would not need to be modified or redesigned. Ideally, this project will serve as a basis for future SAR research that will move the medical industry a step closer towards normalizing SAR systems as an extension of the clinician's influence into the home setting for the administration of rehabilitation therapy.

Chapter 2

BACKGROUND

2.1 The Need for Rehabilitation

Advances in healthcare have led to a drastic increase in life expectancy in developed and developing nations [3]. However, increasing lifespans pose a challenge to the current healthcare framework, with more adults living longer with chronic, or lifelong, health conditions. In some cases, these conditions lead to decreased quality of life; a subset of such conditions that are increasing in prevalence include neurodegenerative diseases. Neurodegenerative diseases are characterized by a decline in neurological function and comorbid impairments including motor, behavioral, or cognitive limitations. According to the American Heart Association, stroke is a leading cause of serious long-term disability in the United States, reducing mobility in more than half of stroke survivors age 65 and over [2]. In addition, diseases such as muscular dystrophy or Parkinson's disease, which affects almost 1 million people in the United States, can severely inhibit motor control in individuals [4]. Overall, the Centers for Disease Control (CDC) estimates that approximately 40.7 million adults in the United States suffer from difficulties in physical function, over half of whom are under 65 years of age [5].

2.2 Telehealth and Rehabilitation

For many of these physically limiting diseases and disorders, the best (and sometimes only) treatment to improve the affected individual's physical functionality is motor

rehabilitation in the form of physical or occupational therapy (PT or OT, respectively) [6, 7]. PT and OT, when administered by trained professionals, generally adapt to changes in user behavior and performance. This rehabilitation typically requires consistent retraining of motor skills through various movement tasks. Unfortunately, PT and OT are time consuming and are typically prescribed only for a limited duration of time for patients to resume basic level function and activities of daily living (ADLs, e.g., bathing, dressing) according to user needs. Because these conditions are chronic in nature, there remains an unmet need in the population for continuous, adaptive rehabilitation therapy.

With the advent of various forms of telecommunication and the 'internet of things' (IoT), physical therapy is increasingly being administered via mobile communication media such as video conferencing [8, 9, 10, 11]. This long-distance form of therapy can save time and money for many health care companies and their respective clinicians who would otherwise be required to meet their patients in person. Implementations of this practice can be generally categorized as telehealth, which utilizes telecommunications to distribute health-related services, or more specifically as mobile health (mHealth), which provides patients with mobile health applications that may not even require clinician interaction. Such applications include custom tools designed for specific health care monitoring via cell phones, tablets, or other personal devices. Examples of these applications are mobile software apps such as 'MDLIVE', 'LiveHealth', and 'Doctor on Demand', which all allow the user to interact directly with a doctor. Other applications such as 'Telehealth by SimplePractice' or 'Spruce' provide a means to easily organize medical information and keep track of appointments [12].

Despite the increasing number of applications used for these purposes, they are generally limited to monitoring of user status on an infrequent basis. Therefore, existing tools suffer from various limitations: systems are open-loop (no participant feedback is

provided); systems do not adapt to changing health status; and systems are not easily integrated into the existing health care framework or clinical workflow. These mobile device applications are therefore insufficient for the complex, real-time, closed-loop process of motor-neurorehabilitation. Such interventions require constant monitoring of user task performance, provisioning of feedback, and adapting to changes in user state.

2.3 Socially Assistive Robotics

In addition to telehealth via phone or video conferencing, a more recent, technology-driven approach to in-home care is emerging in the research domain. Socially assistive robotics (SAR) utilizes (typically embodied) software agents to assist humans through hands-off, non-contact social interaction. SAR is a newer field derived from ongoing research in the domains of human-computer and human-robot interaction (HCI and HRI, respectively). These robots are increasingly being used to extend the reach of clinicians into the ambient or home setting. Matarić defines socially assistive robotics as, "the intersection of [assistive robotics] and [social interaction robotics]. SAR shares with assistive robotics the goal to provide assistance to human users, but it specifies that the assistance is through social interaction." [13]. Though there exists significant variety in the design of such systems, the common components include the human user, sensor modalities capable of measuring user behavior, and the software/hardware agent. Such SAR systems have been used in research and commercial settings to alleviate a range of conditions.

2.3.1 Healthcare for Elderly, Dementia, and Alzheimer's Patients

Socially assistive robots have been utilized in multiple aspects of elderly care, especially to provide companionship and social interaction to individuals who live alone and to provide physical exercise therapy to sedentary individuals [14, 15, 16]. The lack of social interaction that frequently accompanies living alone has been shown to be detrimental to these individuals' physical and mental health. In many cases, this loneliness, paired with conditions such as dementia and Alzheimer's disease, can directly lead to clinical depression [17]. In addition, many older adults benefit greatly from physical therapy to encourage consistent exercise, which has been shown to be effective at maintaining and improving the overall health of older adults [18].

SARs such as Rassle, the active teddy bear robot (shown in Figure 2.1), engage elderly people and respond to tactile stimuli [17]. Rassle was used in a memory game for older adults. Users were prompted to touch parts of the teddy bear in a specified order, having to memorize the touch pattern and perform correctly multiple times at increasing difficulties. The study found that the users were touching the robot approximately 17% of the time throughout the duration of the interactions. The physical and mental actions associate with this process may lead to significant gross motor activities and mental stimuli for these elderly people [17].

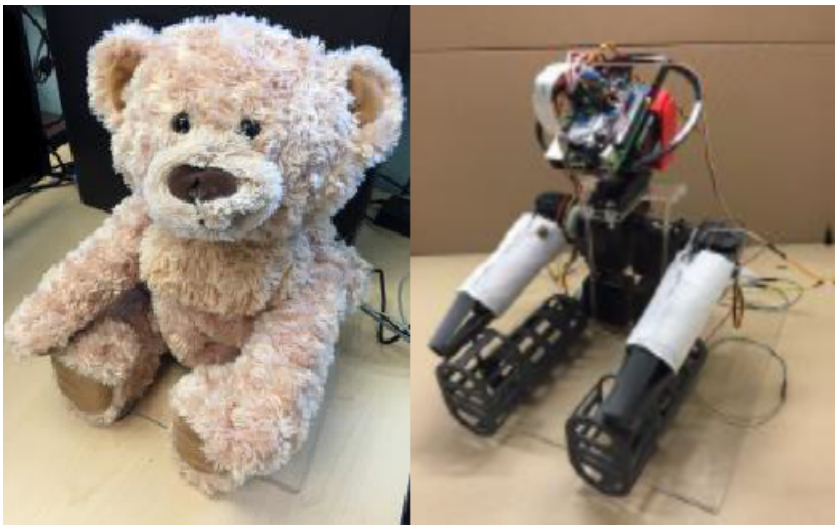

Figure 2.1: The tactile sensing social robot teddy bear, Rassle (left) and its robot skeleton (right). Adapted from [17].

In addition to social interaction, SARs have been used to provide physical exercise therapy to sedentary older adults. Figure 2.2 shows an example of such a use case in which an elderly individual is being instructed to perform simple exercises [18]. Throughout a session, the robot prompts the user to perform simple seated arm gesture exercises. This type of seated exercise is commonly practiced in senior living facilities for individuals with low mobility [18]. The robot acts autonomously without the need for human intervention and the user can interact directly with the robot via the button interface of a Bluetooth Wii remote.

Figure 2.2: An exercise setup with user and robot facing each other. Adapted from [18].

This SAR system was used in a study in which 24 individuals were encouraged to perform actions from three different game categories: workout, imitation, and memory [18]. In all cases, the feedback from the users was positive with respect to their perception of the system.

2.3.2 Mobility Training for Infants

Another relatively unexplored application of SAR systems is mobility training for infants with developmental limitations [19, 20]. An estimated 9% of infants born in the United States are at risk of underdeveloped motor control and strength due to a lack of movement or other developmental delays [21]. These infants could benefit from early intervention to elicit extra movement that would improve this development.

Figure 2.3 shows an example of an infant interacting with a SAR system that consists of a humanoid robot with multiple sensors [21]. This system was developed

to compliment human-delivered therapy for infants that would benefit from early intervention.

Figure 2.3: An experimental setup in which an infant interacts with an NAO robot while the labeled sensors (an eye tracker and inertial sensors) and additional sensors (RGB cameras and a Kinect One RGB-D sensor, which are not shown in the field of view of this image) capture information about the infant–robot interaction. Adapted from [21].

Twelve six- to eight-month-old infants participated in this study that utilized different feedback methods to encourage leg movement [21]. These feedback methods were meant to encourage the infants to imitate the robot's movement and move beyond their normal range of motion. The study determined that the infants would move more when trying to imitate the robot than they normally would when the robot was stationary. This positive result further reinforces the value of this type of SAR system in physical therapy applications.

2.3.3 Treatment of Children with Autism Spectrum Disorder

SARs have additionally been used to help children with autism spectrum disorder who require special instruction in social behavior and emotional aspects. Humanoid robots, such as Milo in Figure 2.4, are often used to interact with children as friendly companions that teach various behavioral lessons.

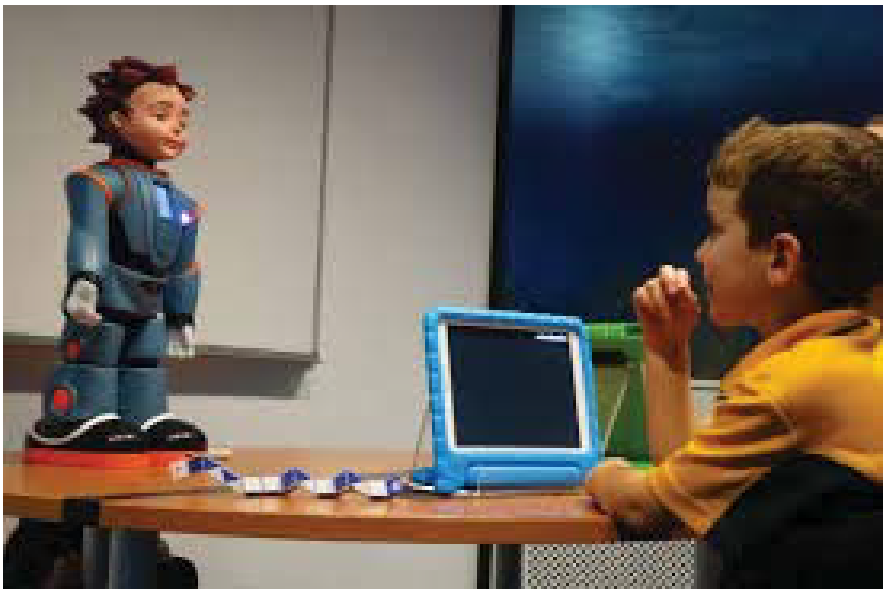

Figure 2.4: The humanoid robot, Milo, developed by RoboKind. It can walk, talk and model human facial expressions. It delivers lessons verbally to teach social behavior and emotional aspects. Adapted from [22].

SARs such as these have successfully been used to help children with autism spectrum disorder, inspiring further research into applications of these systems for various conditions.

2.4 The Need for a SAR Framework

In all cases, socially assistive robots are used to facilitate medical care that is directly mapped to existing human-based interventions. These SAR tools are typically de-

signed to extend the limitations of intervention timing or treatment location (e.g., the patient's home) beyond the capability of the current healthcare model. If this practice were to become commonplace, SARs have the potential to save large amounts of time and money for health care companies that can then better utilize their clinicians for more essential medical cases.

Despite this potential, many questions regarding the design and implementation of such systems remain, including the role of embodiment, the nature of feedback provided to users, and how best to design robot-based interventions. As socially assistive robots become more prevalent, it is necessary to develop tools capable of systematically evaluating properties of SAR systems and their interactions with user performance. Existing uses of SARs utilize software for each robot that is specifically designed for a certain rehabilitation task, tracking a pre-specified performance metric and providing feedback to the patient. While this approach works well on a case-by-case basis, a more generalized software framework that allows for customizability and modularity will be beneficial for testing and design of SARs, and for the systems that will ultimately be deployed for many different types of rehabilitation tasks. Being able to easily change the robot's interaction parameters, such as the feedback type or performance metric, would make such a system more versatile for various medical applications. The design and implementation of this modular human-robot interaction (HRI) software framework is the focus of this project. If successful, the outcomes of this project will be used in ongoing research and will be made publicly available for other researchers in the domains of SAR and HRI.

Chapter 3

SYSTEM DESIGN

3.1 Introduction

The human-robot interaction framework described in this chapter is a software framework that is developed as a tool to be utilized and adapted for future projects. There are many existing examples of software frameworks with use cases for socially assistive robots; however, they are generally built for use with specific hardware and only support one specific type of motor task [23, 24]. The novelty of the framework design described in this chapter is the generalized modular structure that will allow for the development of a variety of SAR interactions. This framework will provide a standardized structure for such systems in addition to providing support for a variety of hardware interfaces and motor tasks. These future SAR systems will be used for projects that will consist of human-robot interaction in an experimental setting that is conducive to improving human motor performance. The generalized structure of this experimental setting is described as follows:

A human test participant is required to perform a task that exercises their motor control. The results of performing this task must be quantifiable and the human must know the goal of the task and be incentivized to perform as well as possible. Once the task is completed, an external feedback delivery medium, such as a speech module, screen, or robot, provides feedback to the human regarding their performance on the task. The human then continues to perform the task multiple times while receiving performance-dependent feedback. After showing proficiency in the task, the difficulty

of the task may be increased to further test the human's ability. Throughout the experiment, task-specific performance data are collected and logged for analysis.

The framework is designed to facilitate this feedback system, taking input from the human participant, analyzing it, and providing specialized feedback based on predetermined performance metrics. As it is necessary for the system to be easy to understand and modify for any specific task, the focus of the framework design is on its modularity and simplicity. The following are the specifications for this project.

3.1.1 Task-Oriented Training

Task-oriented training (TOT) consists of tasks that are specifically tailored to the user's needs. This type of training is necessary for motor rehabilitation as it utilizes repetition and difficulty scaling that can be adapted to the current state of the user [25, 26, 27]. TOT is meant to counter poor compliance with rehabilitation by building intrinsic (as opposed to extrinsic) motivation, which results in increased engagement and compliance with intervention. TOT generally relies on the performance of meaningful tasks (e.g., lifting a weighted bag of groceries) for which the user can clearly see the benefit in practicing. In order for this aspect of the system to be fully developed, the system must be capable of administering tasks that have three distinct properties within the context of a task-oriented training program.

The system must have a quantifiable outcome and performance metric that corresponds to the task that the user is performing. For example, in one study that aimed to improve upper-limb motor function in elderly patients who had mild post-stroke impairment, multiple quantifiable performance measures were used, including The Upper Extremity Performance Test, the Upper Extremity Fugl-Meyer Assessment (UE-FMA), shoulder flexor and handgrip strength, shoulder active range of motion,

and muscle tone [26]. Though some of these instruments include rote, repeated tasks, the UE-FMA includes tasks that are more closely tied to activities of daily living. These metrics were successfully used in a task-oriented training program to improve the motor functionality of these elderly individuals.

The system must also have a quantifiable level of difficulty that can be changed in response to the user's performance. Researchers have shown that learning occurs when tasks are administered at an optimal difficulty level [25]. When the level of difficulty is too low, no learning occurs. When the difficulty is too high, the user may become frustrated and quit. In the aforementioned study, the greatest improvement was seen in the group of individuals who received personalized resistance training based on their ability and performance [26].

Finally, the system must have a formula with modifiable parameters for changing the difficulty of the task according to the user's performance. The complexity of this formula (e.g., heuristic vs. algorithmic) will depend on the application.

3.1.2 Augmented Feedback

Augmented feedback consists of feedback that supplements the natural feedback that a user receives when doing a task. This feedback can be classified as knowledge of results (KR), in which the user is provided information regarding their achievement of a specified goal, or knowledge of performance (KP), in which the user is informed of the quality of their performance [28, 29]. For this aspect of the system to be fully developed, the system must have a feedback element that has two distinct properties.

It must be possible to configure the system to provide feedback after a specified number of task attempts. Additionally, it must be possible to configure the system to provide this feedback at varying levels of specificity and in different forms, being ei-

ther qualitative or quantitative in nature. A particularly useful study investigated the role of KR and KP (provided by human researchers) for individuals post-stroke learning a pointing task [30]. Participants were required to use their impaired arm to point to a target. The KR group received quantitative feedback in the form of an error value (the Euclidean distance from their finger-tip to the target) while the KP group received feedback on the quality and smoothness of their joint motions during the pointing process. Interestingly, this study found that participants in the KP group demonstrated improved learning. This result suggested that humans preferred more 'human-like' feedback (e.g., qualitative instead of quantitative) from a human therapist. Another relevant study of augmented feedback for velocity training for rugby players showed that the frequency and form of the feedback given had a significant effect on the overall motor performance and retention of the individuals [29]. Individuals who were provided feedback immediately after each exercise tended to perform better overall than those who received delayed or no feedback.

3.1.3 Embodiment

Human-robot interaction studies often place emphasis on the medium through which feedback is provided to the user. The study of this concept of 'embodiment' aims to evaluate the effects of the delivery medium on performance. Studies have generally shown that a more human-like embodiment (e.g., a social robot, as compared to an avatar on the screen) leads to better performance outcomes [31, 32].

To satisfy the use requirements of this framework with respect to possible embodiment, the system must be capable of running on a variety of hardware platforms, including social robots, tablet/screens, and laptops. In each case, the system should be capable of using all mechanisms of communication (e.g., gestures, visualization, audio) available to the hardware. The importance of the embodiment aspect of a

SAR system has been studied in multiple environments and is understood to play a major role in the user's reception of feedback and engagement with the system. One study in which a robot was used to help individuals lose weight and then retain their progress showed that participants who interacted with this 'robotic weight loss coach' would typically track their caloric intake and exercise for twice as long as participants who utilized other methods, such as standalone computer or paper logs [31]. It is also conceivable that some audiences may respond differently depending on context. A child participating with a robot may be more engaged if the robot exhibits 'child-like' features; alternatively, an older adult may limit engagement if the delivery medium is deemed too juvenile. For these reasons, the current system must provide support for a variety of feedback delivery media.

3.1.4 Modular Flexibility

The system must provide modular flexibility such that it is easy to change the type of task that the user is expected to perform. This requires a basic 'template' framework that is robust and structured such that it provides all necessary features for future implementation. Additionally, this flexibility should ensure that only necessary parts of the framework must be modified between implementations, leaving the basic structural elements of the system unchanged. In his analysis of a social robot interaction platform, Michaud concluded that, "Adopting a modular hardware/software design approach facilitates the design of subsystems by allowing the reuse of microcontroller boards and programs. It also facilitates debugging and subsequent designs and extensions of the platform" [33]. This design philosophy is a guiding principle in the design of this human-robot interaction framework.

3.2 Methods

The following section describes the high-level conceptual design of the human-robot interaction software framework. Figure 3.1 shows a visual representation of the framework. The diagram is broken into sections that mimic a basic model of human cognition in a therapeutic interaction.

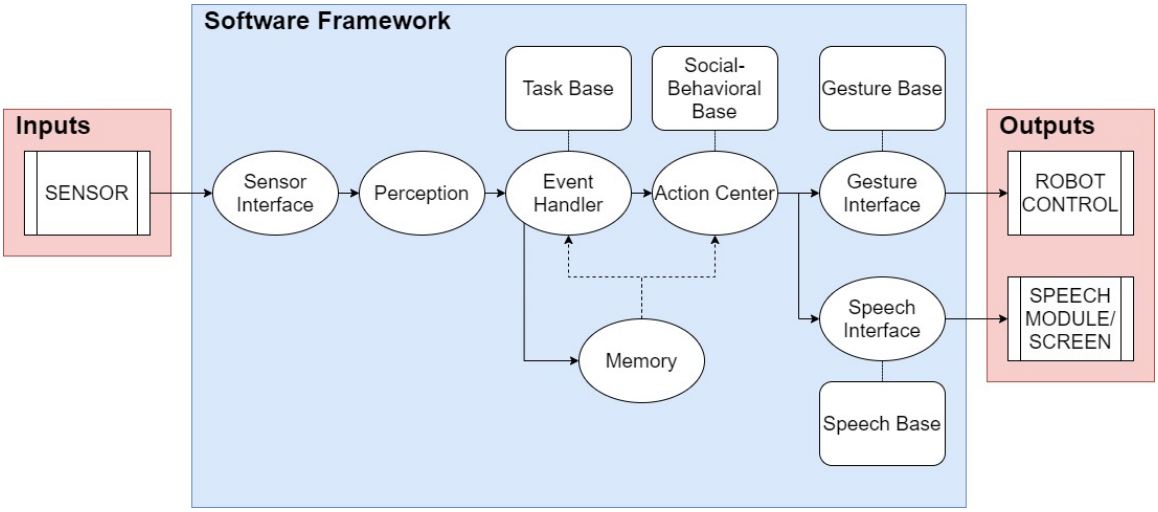

Figure 3.1: Diagram of high level framework design.

In the figure, ovals represent nodes, or individual executable programs that perform a function. Rounded boxes represent bases, or namespaces that contain user-changeable constants and other information that are referenced by the nodes. The Inputs and Outputs of the system represent possible interaction media with the real world. Inputs generally represent physical sensors that interact with the user to collect data and pass them to the system and Outputs represent the feedback mechanisms that interact with the user.

The Software Framework portion of the system exists entirely in software and contains all necessary elements to analyze input data and provide feedback to the user. The Software Framework contains communication protocols between nodes, indicated by

solid and dashed arrows in the figure. Solid arrows indicate conditional direct communication between two nodes that is driven by the node that is passing data. Dashed arrows indicate a conditional request for data from one node by another node, which is prompted by the node that is receiving data. The following sections describe the intended functionality of the different nodes.

3.2.1 Hardware Interfacing: Inputs and Outputs

The framework is designed to interface with hardware for both user input and feedback output. In general, any combination of physical sensors can be used as inputs to the system, acting as the primary data source that drives feedback. With respect to outputs, the system is meant to interface with a screen for visual text output, a speech module for auditory text-to-speech output, and/or motion hardware such as motors for a humanoid robot to facilitate gesture or expression output.

3.2.2 Input Processing: Sensor Interface and Perception

On the software side, every sensor requires its own respective node to interface with the sensor and extract data, called 'Sensor Interface' nodes. These various sensor data can be fused or otherwise combined together in a 'Perception' node. Within this node, the data are filtered and converted into a useful form that can be analyzed by the system, ensuring that only apposite data are used to determine performance. Once the data have been processed they can be passed through the system and used to determine the appropriate feedback type or change in task difficulty. The concept of this sensor-perception input structure is an abstraction of human sensory perception that is commonly used in SAR feedback systems [34].

This design is meant to allow any combination of sensor data to be used in the system, making it possible to have multiple sources of external information that drive feedback. The ability to vary the input data types provides the modularity necessary to implement the framework with any type of task that has a measurable and quantifiable metric of user performance.

3.2.3 Task Flow: Event Handler

Once data are processed they are passed to the 'Event Handler' node. This node controls everything to do with the task that the user is meant to perform, prompting the user to perform certain actions when necessary. It accepts processed data from the 'Perception' node and uses them to determine the user's performance. It can then trigger specific events accordingly, such as increasing task difficulty, informing the user of their performance, or providing other forms of feedback. Increasing task difficulty and calculating user performance are handled within the node, while any triggering of qualitative or quantitative feedback in the form of speech or gesture is passed to the 'Action Center' node. Additionally, user performance data is stored for future conditional use in the 'Memory' node.

The 'Event Handler' node references a Task Base which contains constants and global variables related to the task. This includes parameters to modify the timing of task attempts, the frequency at which feedback is given to the user, and other constants that alter the algorithm to determine feedback. In addition, performance data from previous task attempts can be requested from the 'Memory' node for comparison with the current task attempt to determine the user's progress over time.

The structure of the 'Event Handler' node provides a means of easily changing the task that the user must perform. The only significant differences when adapting

to another task are the data type that is used to determine performance and the subsequent algorithm to determine difficulty and feedback.

3.2.4 Feedback: Action Center and Output Interfaces

Commands from the 'Event Handler' node are received and processed by the 'Action Center' node to provide specific feedback to the user. This feedback can be either qualitative or quantitative in nature and can consist of both visual (text, image, or physical robot gesture) and auditory (speech) feedback types. In this node, the specificity of feedback (qualitative or quantitative) and degree of feedback are determined and used to form the appropriate output. This output is then passed via commands to the output interfaces (speech interface, gesture interface, etc.) which provide the feedback to the user.

The 'Action Center' node references a Social-Behavioral Base which contains constants and other information related to the social nature of the robot. This includes parameters to change the type of feedback from qualitative to quantitative, which is a necessary modular function. In addition, performance data from previous task attempts can be requested from the 'Memory' node to be provided to the user as feedback.

3.2.5 Data Handling: Memory and Data Logging

The 'Memory' node contains data structures for storing any necessary data that may be used by the system. It receives data periodically from the 'Event Handler' node and can send data to the 'Event Handler' or 'Action Center' nodes via specific request. This node provides the means to store data as working or short-term memory and ensures that the other processing nodes do not need to track data.

In addition to storing data as working and short-term memory, the system also contains a logging feature that provides the ability to track any data passed through the system and store the data in an external file as long-term memory. This data can be subsequently imported into MATLAB for analysis.

3.3 Implementation with ROS

The final design of the system implements all of the elements listed above in a single software framework within a middleware suite called ROS (Robot Operating System). ROS is an open-source collection of software frameworks that are designed to make robot software development simple. The ROS structure makes the implementation of a node-based framework easy by providing two built-in types of communication protocols between executable nodes of the framework. A node can either subscribe to a topic on which another node publishes data, or it can send a request to another node which will fulfill the request with a pre-specified service routine.

At the software level, all framework nodes are written in C++, but can be easily modified to use Python as needed for adaptability. All ROS nodes are structured as C++ classes with member variables and functions specific to the node's functionality. Within each node, executable code is triggered when a node receives data from a topic to which it is subscribed. This is implemented via callback functions which act as the main function in each node. When a node receives published data from another node, the corresponding callback function is triggered and the appropriate code is executed to utilize the given data and pass new data to the rest of the system. Utilizing this ROS communication structure, the design of the framework has inherent modularity for interchangeable nodes as well as a large amount of open-source support for plugins and hardware interface libraries.

Chapter 4

SYSTEM IMPLEMENTATION AND VALIDATION

4.1 Introduction

The human-robot interaction framework described in the previous chapter is implemented in two distinct systems for the purposes of this project. The first implementation, which serves as a 'template' for all other adaptations of the framework, is the most basic example of an implementation that meets the first three design requirements: task-oriented training, augmented feedback, and embodiment. The second implementation, which is an adapted form of the first, is the basis for demonstrating the final design requirement of modular flexibility. This chapter describes these two implementations as well as their use in validating the design requirements for the framework.

4.2 Implementation

This section describes the two unique implementations of the framework that are used for testing, validation, and development in future work. Both implementations consist of the same structure built on top of the framework.

4.2.1 Implementation I: Keyboard Input Task

The first implementation is a purely software-based interaction system which consists of a keyboard input task with text output as feedback. Key-pressing tasks have been

used in a variety of populations to measure motor and cognitive performance [35, 36, 37, 38]. Variations to the task (e.g., repeated vs. random sequences; performance of tasks after experimental manipulations such as fatigue) can be used to extract insight on motor learning and control. Typically, these tasks consist of an external stimuli, a keyboard or a set of keys, and reaction time as a performance metric. The user's goal during this task is to react to on-screen prompts by pressing the correct character key on the keyboard as quickly as possible. Figure 4.1 shows a high-level visual representation of the keyboard input task implementation.

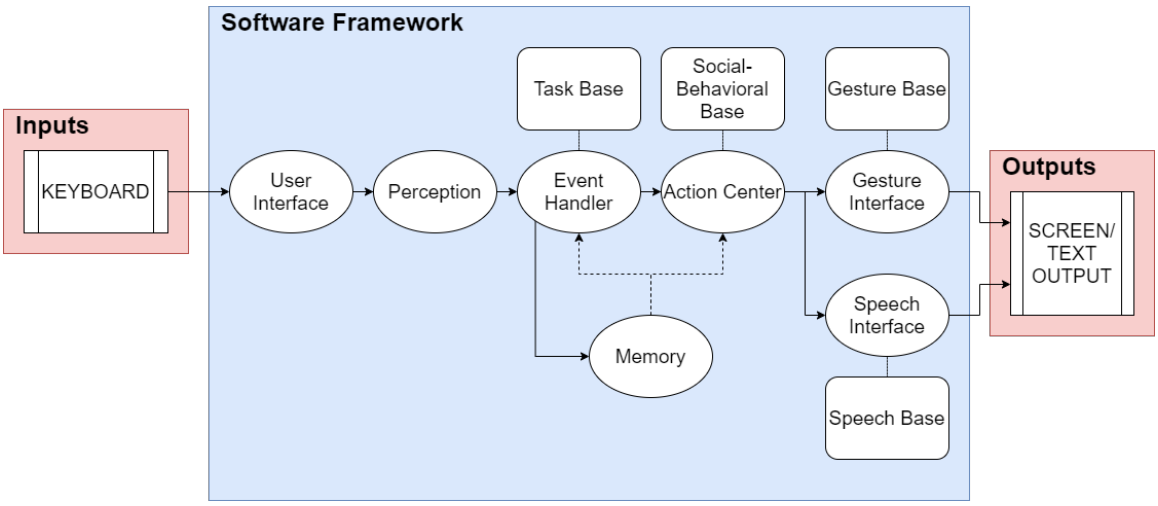

Figure 4.1: Diagram of keyboard input implementation.

Comparing this diagram to Figure 3.1, the structure of the implemented system is identical to the conceptual design of the framework. In this case, the input is recorded directly from the keyboard by the user interface node and all speech and gesture output appears on a screen as text output. Figure 4.2 shows a ROS-generated graph of the implementation as it exists at the software level within ROS.

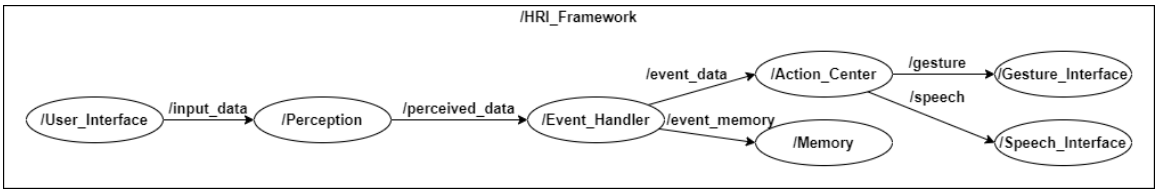

Figure 4.2: ROS-generated graph of keyboard input implementation.

This diagram shows the executable nodes of the framework and the data that is transferred between them. Within the framework, all communication between nodes is executed via subscription/publication protocols except for the 'Memory' node, which works on a server/client basis, fulfilling services as they are requested. The User Interface node reads the user's keyboard inputs and passes them individually to the Perception node where the key that is pressed and the time between key presses are recorded and sent to the Event Handler node as a single data structure (perceived_data). The Event Handler then reads and writes data to and from the Memory node while also passing commands (event_data) to the Action Center node to trigger feedback. The structure of this diagram matches the conceptual design from Figure 4.1, which indicates that the system's structure is successfully implemented at the software level with respect to its intended design.

In this system, each individual key press is sent through the system independently to trigger an event. At the start of the task, pressing the return key will begin the next task attempt. Once the attempt begins, each subsequent key press is compared to the expected key that is prompted from the user to determine if they pressed the correct key. The total elapsed time between each on-screen prompt and the subsequent key press from the user and the accuracy of each key press are used to determine reaction time performance for each task attempt.

Each attempt consists of a specific number of prompts (twenty by default) that are issued to the user, from which an average overall score is derived based on the average

reaction time of key presses and the number of correct key presses. This performance metric is compared to a baseline value to determine if the difficulty should be changed and what type of feedback (positive, negative, or neutral) should be given for that attempt. Figure 4.3 shows an example of quantitative feedback that is given to the user after a successful attempt.

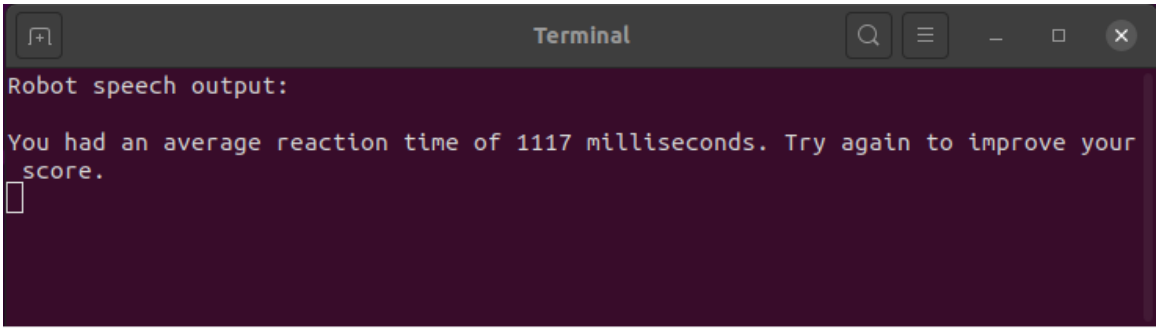

Figure 4.3: Example of quantitative feedback given to the user.

The number of consecutive successful or unsuccessful attempts necessary to cause a difficulty change, as well as the frequency that feedback is provided to the user, can be manually changed as a parameter of the system. Figure 4.4 shows data taken for a typical average user after twenty task attempts.

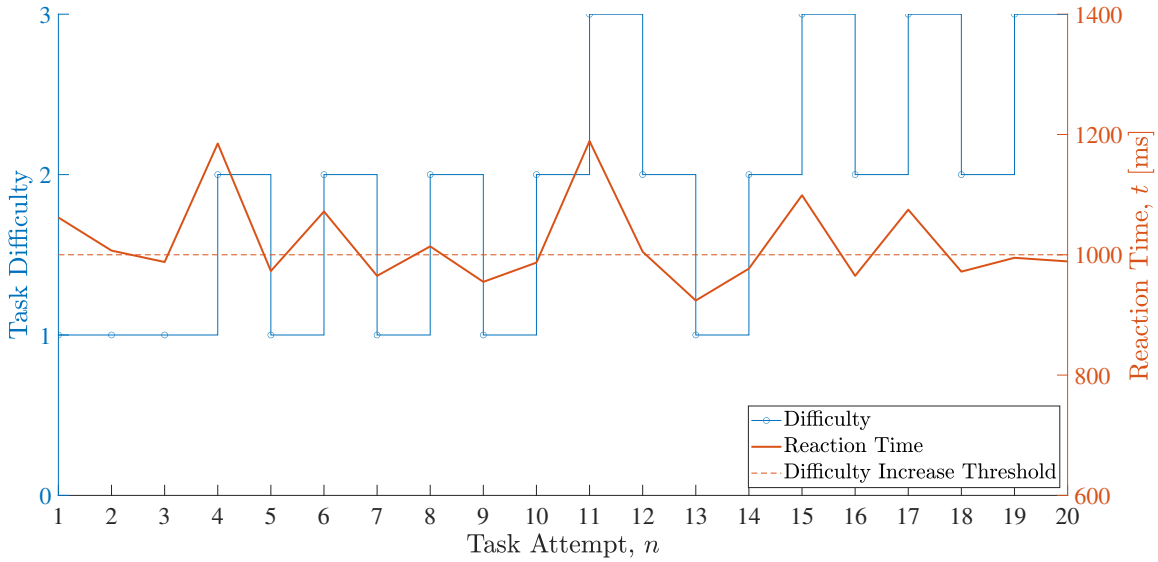

Figure 4.4: Changing task difficulty for average user over twenty task attempts.

Over the course of the twenty task attempts shown in the figure, the difficulty is changed whenever the user's average reaction time is above or below the threshold (1000ms in this case). When the user's reaction time is faster than 1000ms the difficulty is increased. As would be expected, the difficulty increase causes a worse performance on the subsequent attempt, causing the difficulty to be decreased again. To prevent the repeated increase and decrease of task difficulty, the system can be configured to require a certain number of consistent attempts above or below the threshold before the difficulty is changed.

4.2.2 Implementation II: Reaching Task

The second implementation builds on the first by utilizing an inertial measurement unit as hardware input in addition to the keyboard input of the first implementation. For this adapted system, the user's goal is to reach their hand from a predetermined starting position, touch a specific marker, and return their hand as quickly as possible.

This task is an example of 'goal-directed reaching,' which requires the user to move the arm towards a target or an object within a specified amount of time [39, 40, 41]. Such tasks have been particularly useful in quantifying impairment in individuals post-stroke [42, 43, 44]. Figure 4.5 shows a visual representation of the reaching task implementation.

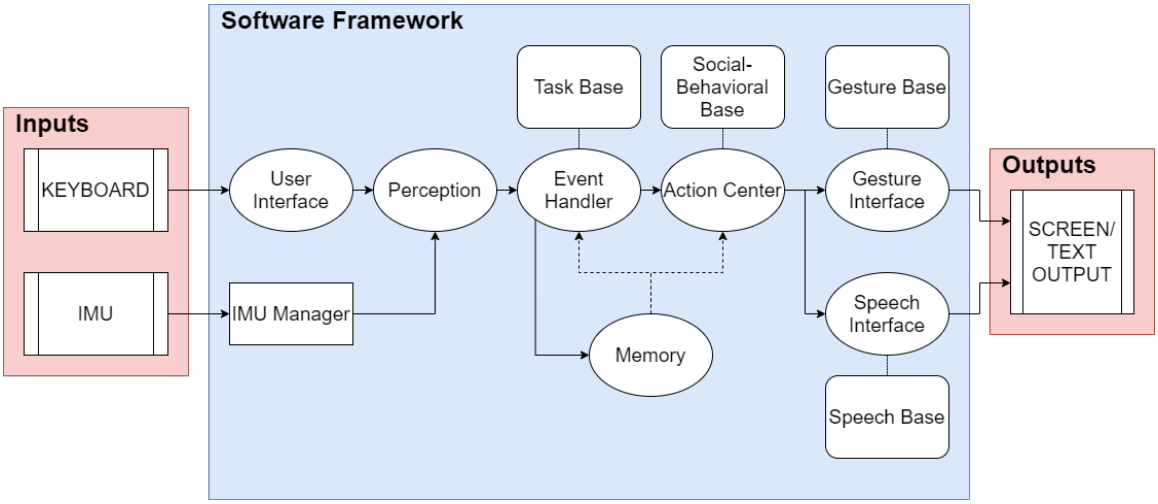

Figure 4.5: Diagram of reaching implementation.

The structure of the system matches that of the keyboard input task implementation, with the addition of IMU input data. The IMU used in this system is the PhidgetSpatial Precision 3/3/3 9-axis IMU (ID: 1044_1B). This hardware input is managed by a new system (the IMU Manager) that constantly polls the IMU and extracts data to be sent to the rest of the system. Figure 4.6 shows a ROS-generated graph of the implementation as it exists at the software level within ROS.

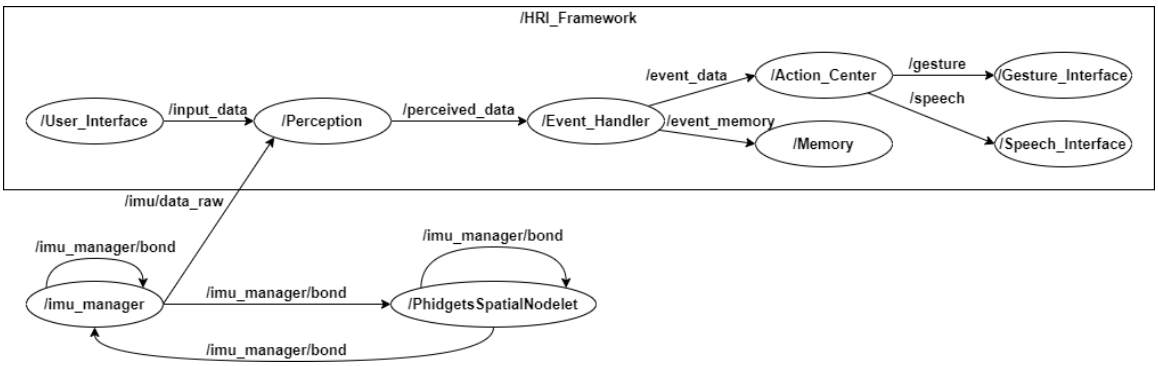

Figure 4.6: ROS-generated graph of reaching implementation.

This diagram shows that the framework contains executable nodes that match the intended conceptual design of Figure 4.5. As with the keyboard input task, all communication between nodes is executed via subscription/publication protocols except for the 'Memory' node, which works on a server/client basis, fulfilling services as they are requested.

For this task, the user is outfitted with a wrist brace that secures the IMU to the back of their hand with the y-axis aligned parallel to their fingers and the z-axis aligned outward from the back of their hand as shown in Figure 4.7. In this system, key presses from the keyboard are only used to mark the beginning and end of a reaching motion. Similar to the keyboard input task, pressing the return key begins the next task attempt. Once the attempt begins, the user presses the spacebar key with their free hand when they are ready to begin their reach. At this point, the user reaches their other hand as quickly as possible from a marked starting position to a marked ending position some distance away from them, then back to the starting position. Once the reach is complete, the user presses the spacebar key again with their free hand to end the reaching motion.

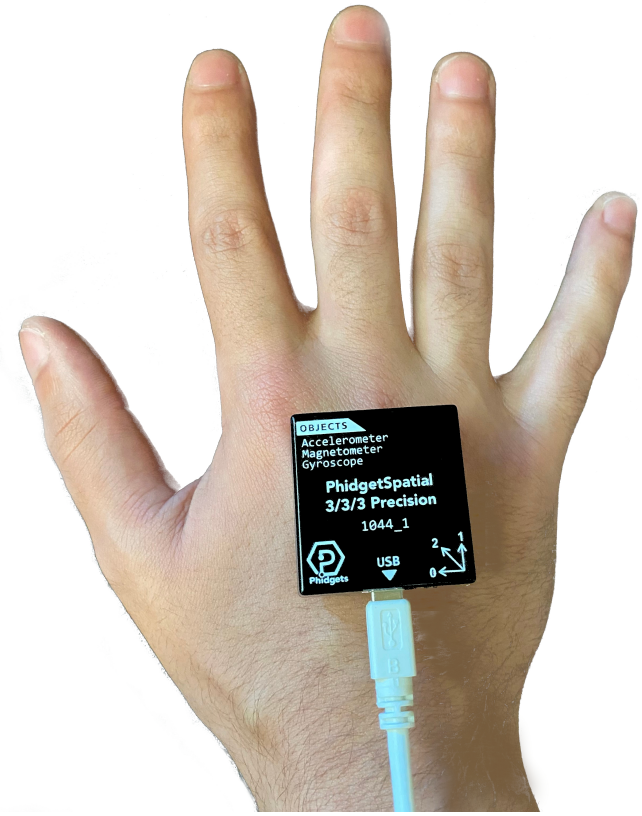

Figure 4.7: Image of IMU positioning and orientation on the back of a hand.

After each reach the system analyzes the data collected from the IMU to determine the total time of the reach by detecting the beginning and end of the motion. The beginning of motion is marked by angular velocity of the wrist in the z-axis that exceeds a specified movement threshold (0.1rad/s by default). The end of motion is marked by a period of no motion, or motion below the movement threshold, that lasts for a certain period of time (800ms by default). These two time marks are used to determine the total reach time, which is used as the performance metric to determine difficulty changes and feedback in the same way as the keyboard input task. Figure 4.8 shows example angular velocity data about the z-axis from the IMU's gyroscope for different types of reaches performed in a test of the system. This data, taken from

the author, Michael Juri, when piloting the system, is representative of the data that the system receives from the IMU.

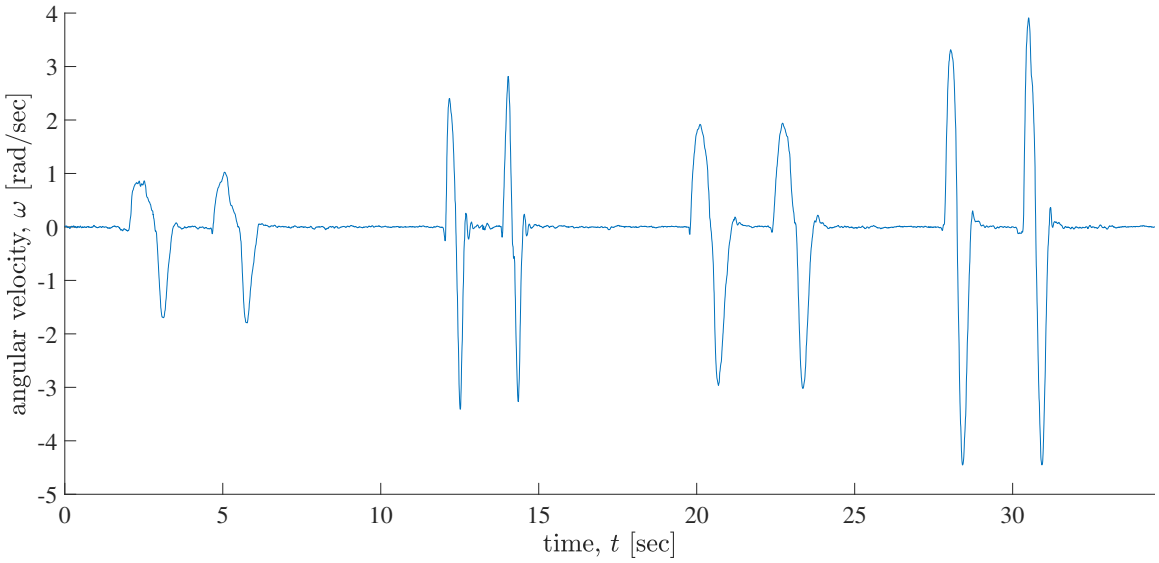

Figure 4.8: Angular velocity data about the z-axis for varying reach types.

The reaches performed in the test consist of four different types, performed twice each for a total of eight reaches. These four reach types, in order, are a slow-moving reach to a near target, a fast-moving reach to a near target, a slow-moving reach to a far target, and a fast-moving reach to a far target. Each reach is depicted in the figure as an elapsed period with a single large positive spike followed by a large negative spike. Table 4.1 shows the reach time and amplitude for each reach type.

Table 4.1: Reach time and angular velocity amplitude for varying reach type.

Reach Type	**Reach Time [ms]**	**Amplitude [rad/s]**	
		Extension	**Return**
Slow, Near	1.3385	0.8622	-1.6876
	1.4416	1.0226	-1.7875
Fast, Near	0.8117	2.4040	-3.4104
	0.8239	2.8159	-3.2666
Slow, Far	1.4920	1.9187	-2.9641
	1.5169	1.9366	-3.0210
Fast, Far	1.0922	3.3147	-4.4224
	1.0792	3.9105	-4.4508

As seen in the table, the slow reaches had a greater reach time, but lower amplitude compared to the respective fast reaches of the same distance. This difference in total reach time and maximum velocity is an expected outcome as a fast reach to a target would be expected to take less time, but require greater velocity. In addition, the data show that reaches to farther targets incurred a higher amplitude than reaches to closer targets. This difference in maximum velocity is also expected as a larger reach distance will naturally draw a greater movement velocity due to the necessity of making the movement as quickly as possible.

4.3 Validation

For both implementations of the framework, validation of design requirements consists of stress and feature testing of each system separately. To consider the project a success, the first three design requirements (task-oriented training, augmented feedback, and embodiment) must be satisfied by the the keyboard task implementation. Additionally, inspection of the reaching task implementation must indicate sufficient modularity of the framework to satisfy the final requirement of modular flexibility.

When testing to validate these requirements, the systems are run for two types of users. The first user represents a poorly performing user while the second user represents a competent user. For the first two requirements, task-oriented training and augmented feedback, validation is dependent on the system reacting properly to both types of users. For all other requirements, validation is dependent on inspection and analysis of the system's structure and features.

4.3.1 Task-Oriented Training

Validation of the task-oriented training requirement consists of functional testing of the first implementation of the framework. In general, this system and any subsequent implementations should follow the same task flow structure, and thus would be considered a task-oriented training system. To confirm that the framework meets this requirement, it must be shown via testing that the system has:

1. A quantifiable outcome and performance metric
2. A quantifiable level of difficulty that can be changed
3. An algorithm for increasing the difficulty

To test these requirements, the system is run once with each type of user for twenty consecutive task attempts. For the poorly performing user, the system would be expected to keep the user at lower task difficulties as their performance should not often meet the requirements to be considered sufficient for higher difficulties. For the competent user, the task difficulty would be expected to increase as the user performs above the threshold of success at each difficulty. Throughout the trial, the reaction time and task difficulty are recorded for each attempt in the same format that is shown in Figure 4.4. For the task-oriented training requirement to be validated, the data must show that the system modifies the level of difficulty based on the user's performance.

4.3.2 Augmented Feedback

Validation of the augmented feedback requirement consists of functional testing of the first implementation of the framework. In general, this system and any subsequent

implementations must provide feedback to the user based on their performance. To confirm that the framework meets this requirement, it must be shown via testing and inspection that the system can:

1. Provide feedback after a specified number of task attempts
2. Provide feedback at varying levels of specificity, being either qualitative or quantitative in nature

To test this requirement, the system is run once with each type of user for twenty consecutive task attempts. For both types of users, the system would be expected to provide feedback indicating that the user's performance is above, below, or approximately average. The poor user would be expected to receive negative feedback more often as their performance is consistently inadequate, while the competent user would be expected to receive positive feedback more often. Throughout the trial, the reaction time and feedback type (positive, negative, or neutral) are recorded for each attempt. For the augmented feedback requirement to be validated, the data must show that the system provides the appropriate type of feedback based on the user's performance.

In addition to the above testing procedure, it must be shown by inspection that the system can switch between qualitative and quantitative feedback. Qualitative feedback should indicate the quality of the user's performance without specifying anything related to the result of the task, while quantitative feedback should indicate the user's performance including providing information regarding the result of the task. Specifically for the keyboard input task, qualitative feedback may specify whether the user is doing well or poorly for each task attempt, while quantitative feedback may specify the average reaction time of the user on the previous attempt relative to past attempts to indicate progress or regress.

Finally, it must also be shown that the system can provide feedback at a specified frequency, independent of the type of feedback. If the system can be shown to provide feedback according to user performance in this manner, this would confirm that the system properly administers augmented feedback, satisfying all aspects of the requirement.

4.3.3 Embodiment

Validation of the embodiment requirement consists of inspection and analysis of the structure of the framework, its existing features, and possible additional features through future implementation. It must be shown that the framework can run on many different platforms, including social robots, tablets/screens, and laptops. Additionally, it must be shown that the framework has support to interface with various types of hardware to provide feedback to the user, including motion output via motors or other actuators, audio output such as sounds or speech, or text output on a screen. For the embodiment requirement to be considered validated, it must be shown that all of these features are supported by the structure of the framework.

4.3.4 Modular Flexibility

Validation of the modular flexibility requirement consists of inspection and analysis of the reaching implementation of the framework as an adaptation of the keyboard input implementation. The validation of this requirement requires that the adapted implementation be structurally similar to the keyboard input implementation. It must be shown that modifications to the keyboard input task are only necessary for elements of the new implementation that are inherently unique. This would indicate that the modularity of the framework allows for new implementations to retain the

same structure, proving that the framework is flexible and therefore validating the requirement.

4.4 Results

Validation of the aforementioned requirements are presented in the following sections. Due to limitations imposed by COVID-19, testing focused only on pilot evaluation of system capability. All human interaction with the system was performed by the author, Michael Juri, with author behavior being altered to simulate poor and competent users. Table 4.2 summarizes the framework requirements and the means by which they were validated.

Table 4.2: Summary of framework design requirements.

Requirement	Motivation	Validation
Task-Oriented Training	Quantifiable performance; quantifiable difficulty; difficulty algorithm	Testing of Keyboard Input Task
Augmented Feedback	Provide feedback (variable types and frequencies)	Testing of Keyboard Input Task
Embodiment	Multiple delivery media	Analysis of Keyboard Input Task
Modular Flexibility	Multiple task implementations	Analysis of Reaching Task

4.4.1 Task-Oriented Training

Figures 4.9 and 4.10 present the recorded reaction time and task difficulty data for the poor and competent users respectively.

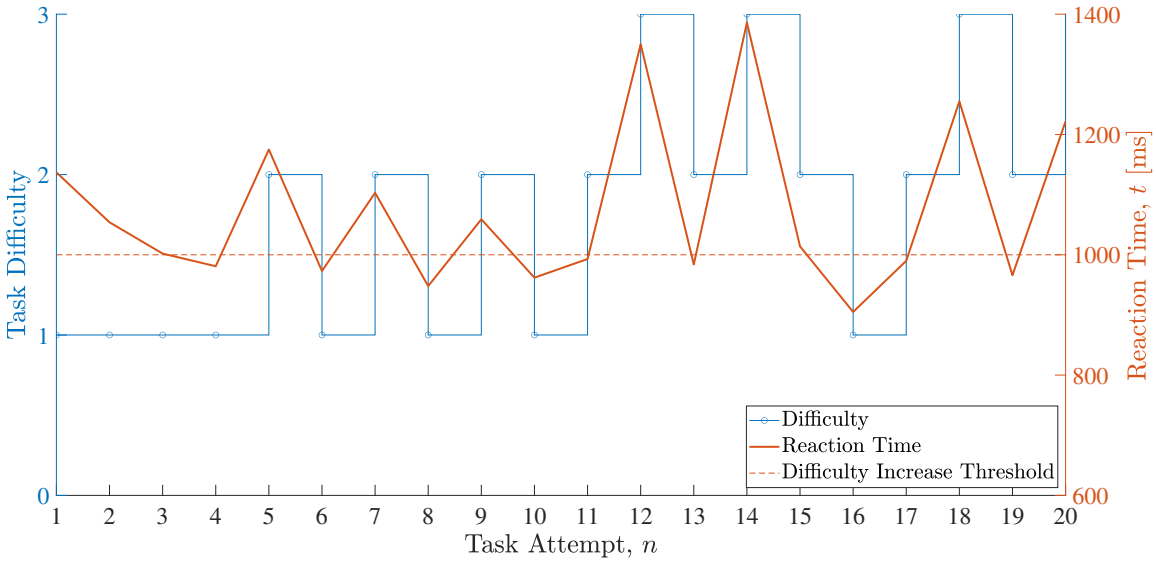

Figure 4.9: Changing task difficulty for poor user over twenty task attempts.

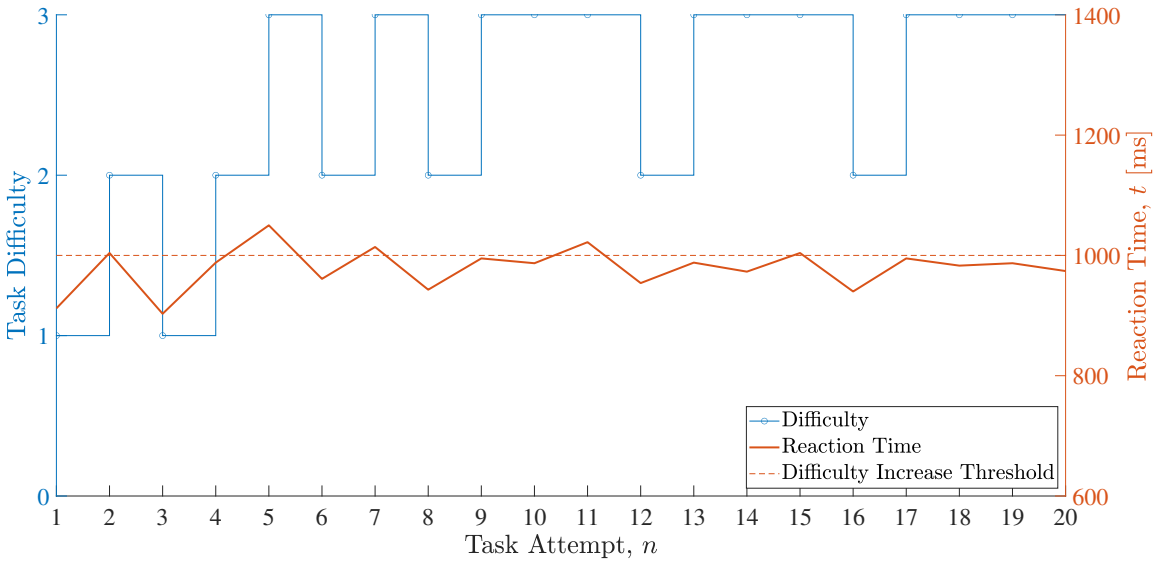

Figure 4.10: Changing task difficulty for competent user over twenty task attempts.

In these trials, when the reaction time of the task attempt (shown in orange) was below the predetermined threshold (in this case, 1000ms), the difficulty of the task was increased by one level (shown in blue). When the reaction time was above the threshold, the difficulty was decreased by one level As expected, when compared to

the competent user, the poorly performing user tended to remain at lower difficulties for longer and found it more challenging to perform consistently at higher difficulties.

From the figures, it is clear that the system modifies the difficulty of the task according to the user's performance. This indicates that the system successfully meets the task-oriented training design requirement.

4.4.2 Augmented Feedback

Figures 4.11 and 4.12 present the recorded reaction time and feedback data for the poor and competent user respectively.

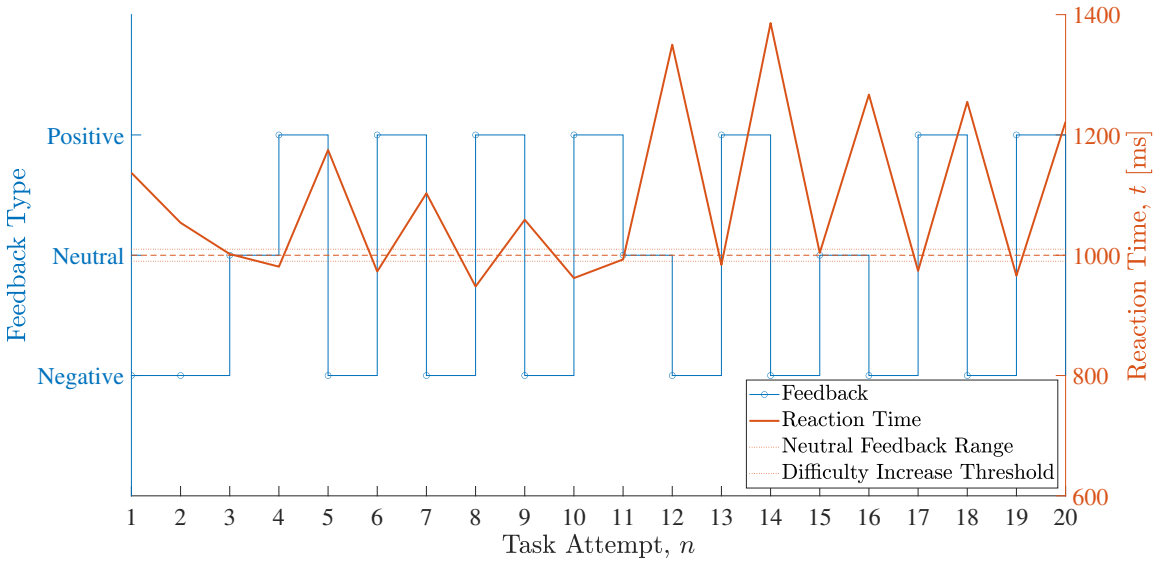

Figure 4.11: Augmented feedback for poor user over twenty task attempts.

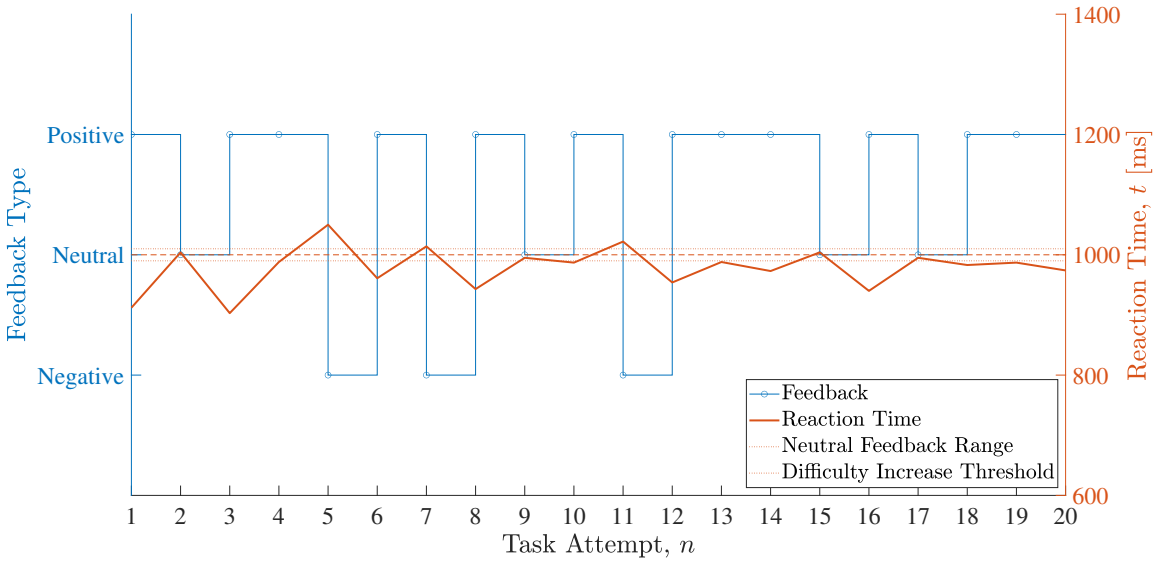

Figure 4.12: Augmented feedback for competent user over twenty task attempts.

In these trials, when the reaction time of the task attempt was below the threshold, the user received positive feedback. When the reaction time was above the threshold, the user received negative feedback. If the reaction time fell within ±10ms of the reaction time threshold, the user received neutral feedback. As expected, when compared to the competent user, the poorly performing user tended to receive negative feedback more often due to the inconsistency of their performance.

From the figures, it is clear that the system provides the appropriate type of feedback according to the user's performance. Additionally, inspection and use of the system shows that the framework provides the ability to provide both quantitative and qualitative feedback at any frequency. Qualitative feedback that is provided to the user consists of a simple statement of poor, adequate, or excellent performance (i.e., the attempt was not satisfactory). Quantitative feedback consists of a statement of performance relative to past attempts, with a numeric comparison of reaction times (i.e., reaction time was 30ms faster than the previous attempt). Examples of these types of feedback given to the user by the system are shown in Figure 4.13.

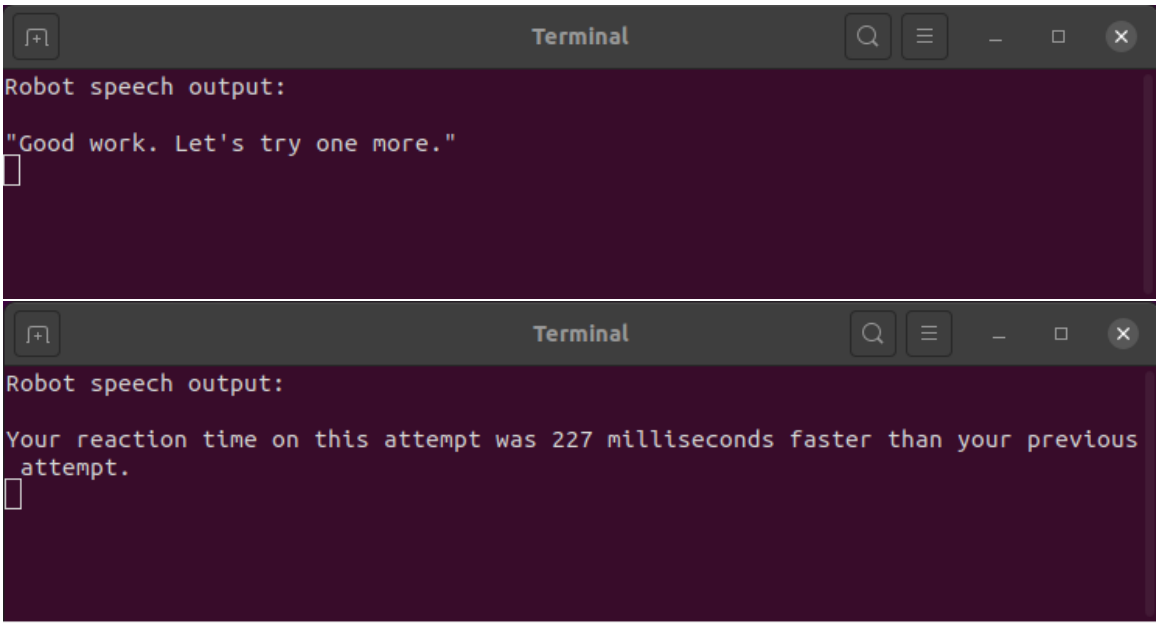

Figure 4.13: Examples of qualitative (top) and quantitative (bottom) feedback given to the user.

Finally, the system was successfully run multiple times with different feedback frequencies. The feedback was limited to be provided only after a specific number of task attempts (three attempts in this case). Figure 4.14 shows a comparison between feedback received every attempt and every three attempts.

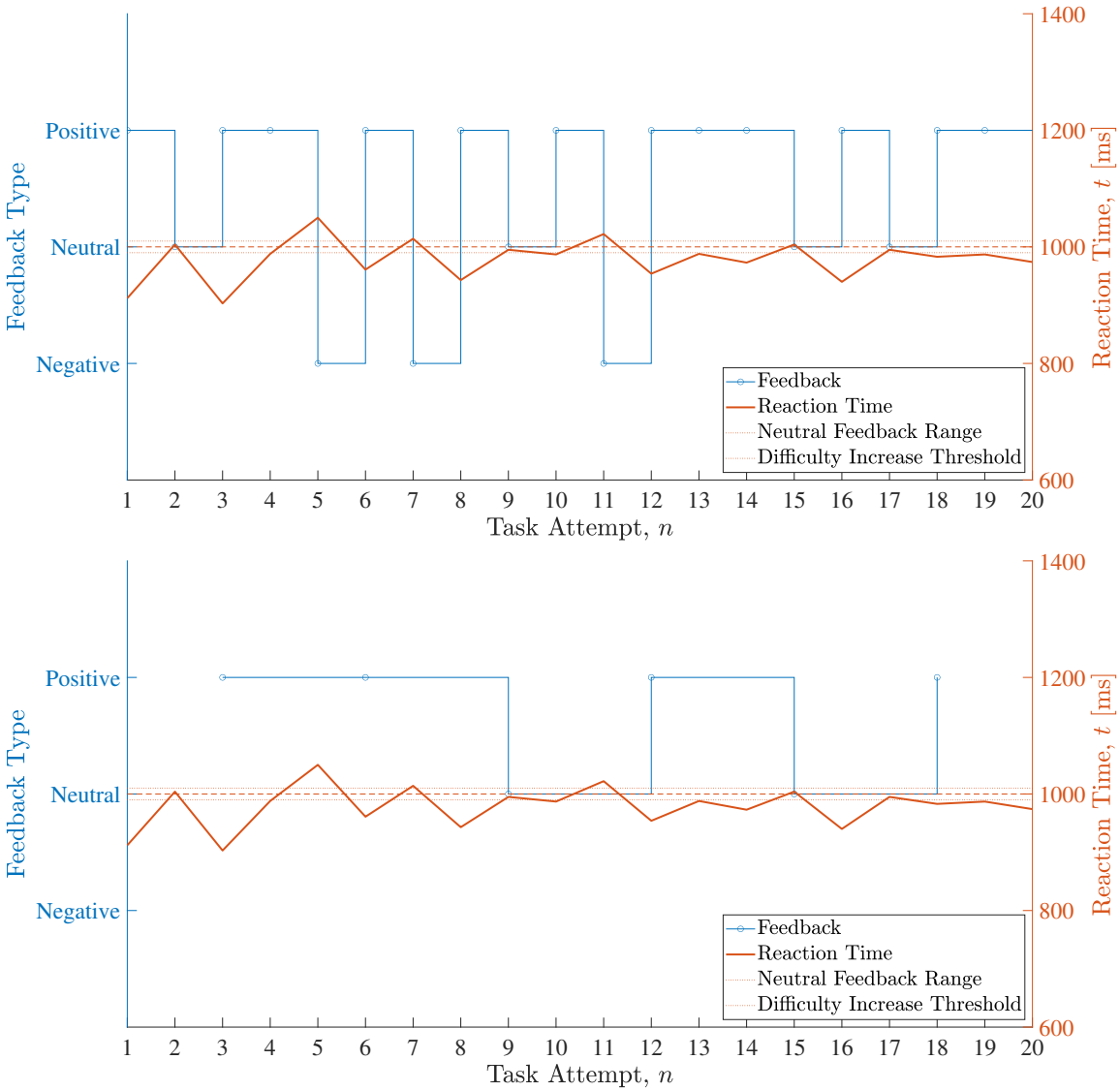

Figure 4.14: Difference in augmented feedback frequency for competent user over twenty task attempts. The top figure shows feedback given every attempt while the bottom shows feedback given every three attempts.

All of these results indicate that the augmented feedback design requirement is successfully validated.

4.4.3 Embodiment

As of this project, text output on a laptop screen is the only feedback medium that is implemented with the framework. Despite this, ROS has many open-source libraries and packages for various devices, including text-to-speech modules, motor drivers, and other robot drivers for feedback output and sensor drivers for data input. As ROS has inherent support for these features, the framework includes this support as it utilizes the ROS middleware suite. Therefore, the embodiment design requirement for the framework is validated.

4.4.4 Modular Flexibility

As shown in the implementation section of this chapter, the structure of the adapted reaching implementation is very similar to that of the keyboard input implementation and, by extension, the framework itself. This indicates that the framework provides an inherent structure that is easily adaptable for different systems. In developing the reaching system, only the Event Handler and Memory nodes were significantly modified from the keyboard input task to accommodate the new data from the IMU. Additionally, an external library was added to the system to handle data collection from the IMU. This library acts as the Sensor Interface node for the IMU and manages the data that is exported by the sensor. Figure 4.15 shows the parts of the system that required modification or addition.

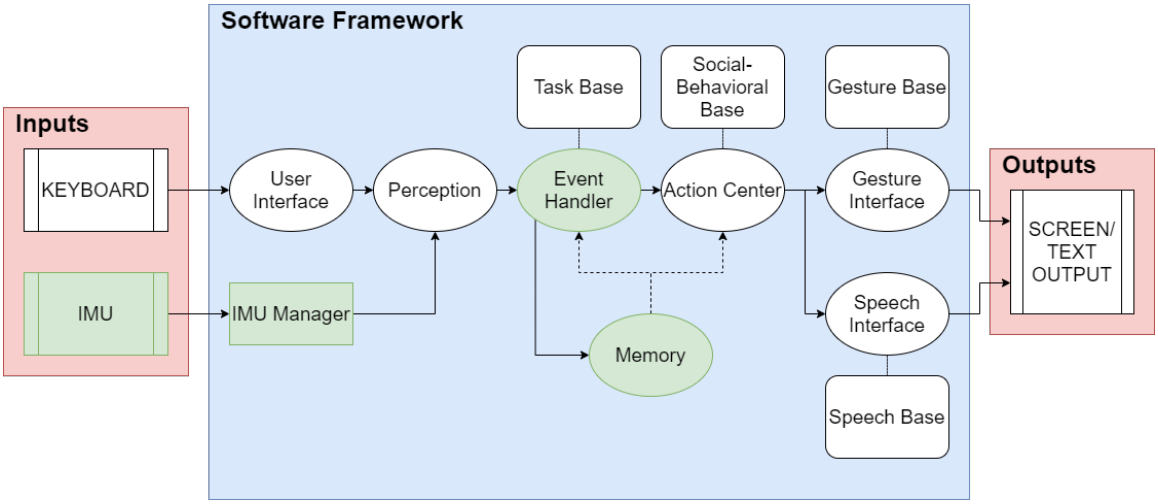

Figure 4.15: Visualization of the aspects of the reaching task implementation that were changed from the keyboard input implementation. All elements of the system that were added or changed are shown in green.

In general, the framework is structured in such a way that the only necessary modifications between systems, barring additional features that may be added on a case-by-case basis, are the data types passed through the system, the determination of the performance metric for difficulty scaling and feedback, and the algorithm for analyzing the data and comparing the performance metric to the appropriate baseline. This consists of an estimated 20-30% of the total code that makes up these systems, indicating that the other 70-80% of the code in the framework is used in every implementation and therefore does not require modification or rewriting. However, depending on the type of system that is being developed, there may be more or less content that must be modified. A thorough discussion of these possible changes will be provided in the next chapter. For the purposes of this project, the ease with which the reaching implementation was developed from the keyboard input implementation validates the modular flexibility requirement.

4.5 Conclusion

In this chapter, the two implementations of the human-robot interaction framework were discussed and shown to share the same underlying structure. These systems were used to successfully validate each of the design requirements through testing and analysis.

Chapter 5

DISCUSSION

5.1 Summary of Findings

The objective of this project was to develop and test a modular software framework that can be used for motor-rehabilitation research and the development of SAR systems. Three functional design requirements (support for task-oriented training; delivery of augmented feedback; support for embodiment) and a qualitative design requirement (modular flexibility) were used as constraints on and guidelines for the framework. The high-level conceptual design was modeled after a basic model of human cognition in therapeutic interactions. This design was implemented using the ROS middleware suite which provided convenient tools for developing the individual nodes of the framework and the communication protocols between these nodes. The framework was then used to develop two implementations, a keyboard input task system and a reaching motion task system. These implementations were used to successfully validate the design requirements through testing and analysis.

5.2 Developing Novel SAR Systems

Two examples of full implementations of the human-robot interaction framework were developed in this project. Though they are based on existing motor task paradigms, the implementations serve as templates for the development of new systems for future research. They contain the minimum required set of elements for a SAR system: a functional task, task-dependent feedback, and a feedback delivery mechanism. Mod-

ifications of these elements for future implementations will depend on the relevant study population. The source code and documentation for both of these implementations can be found online at `https://bitbucket.org/juri017/hri_framework`.

When developing an SAR system from the base framework, the developer can take advantage of the modular structure and select which nodes to include and modify. The framework is structured such that any aspect of the software can be rewritten by a developer with general object-oriented programming experience and only a small amount of necessary knowledge of ROS and the nodal structure of the framework. For example, the algorithm that determines performance-dependent progression of task difficulty can be modified within a single member function of the Event Handler node. This modification exists only within this node and does not affect the functionality of any other part of the system. In this way, each node of the system can be developed and modified independently, which provides the ability to interchange different versions of a node to alter node-specific behavior.

Despite the modular nature of the framework, there are certain elements that must be modified between different implementations if a new task is introduced. These necessary modifications are generally dependent on the type of input data used in the system or the type of task that is administered. Modifications include:

1. Sensors and Peripherals: Novel sensor inputs (e.g., a webcam, IMU, etc.) require a ROS-compatible interface in conjunction with the Sensor Interface node of the framework. This interface must extract raw data from the sensor and feed them to the system. ROS has many open-source libraries that contain support for various types of peripheral interfaces; however, it may be necessary for a new ROS driver node to be written for a specific sensor if this is not the case.

2. Sensor Data and Signal Processing: If input data from the peripheral require processing (e.g., filtering, modification, fusion) the Perception node must be modified to provide this functionality. The role of this node should always be to provide the rest of the system only with the data needed to analyze user performance and provide feedback. Therefore, the Perception node may be trivial (simply passing input data along to the system), or complex (fusing data from multiple sensors into a single data structure to be analyzed by the system) depending on the nature of the task. Incorporation of more complex data processing (for instance, machine learning or statistical inference) may be incorporated into the Perception node.

3. Data Types: Data type(s) used by the system depend on the output of the Perception node. Inter-node communication protocols may need to be modified accordingly. For example, if the Perception node fuses sensor data into one data structure that contains a combination of integers, strings, and vector structures, the Event Handler node will need to be modified to receive the data structure.

4. Feedback Media: The example implementations developed for this project provide feedback via on-screen text. To use alternative media (e.g., a humanoid robot that gestures, a speech module, etc.) or wireless communication (e.g., Bluetooth communication to a mobile phone, Wifi communication to a Raspberry Pi), the developer must utilize or create a novel hardware driver. As was the case with sensor peripherals, ROS provides drivers and libraries for a variety of hardware interfaces, including single-board computers and embedded systems, smart devices, and a variety of commercial robots. Changing the feedback medium will require modification of the Output Interface nodes (e.g., Speech Interface, Gesture Interface, etc.) to accommodate the hardware output accordingly.

5. Other Modifications: All task-specific constants, variables, or functions (especially functions related to calculating performance or determining task difficulty) must be modified to match the new task type. For example, a button-pushing reaction time task will calculate user performance differently than a task using analog performance metrics. These modifications will exist primarily in the Event Handler node.

5.3 Discussion of Existing Frameworks and Libraries

There are many examples of existing SAR systems that incorporate all of the elements that this framework provides [23, 24]. However, these software frameworks lack the modularity and generalized structure that is necessary for easy modification of the system with different hardware or feedback types. Despite this, these systems can be referenced as good examples of complete SAR systems.

In addition to independent SAR systems, HRI researchers are continuously developing more tools for use in SAR systems. One such tool is the "Social Behavior Library", which is a ROS library that is currently being developed by Edward T. Kaszubski at University of Southern California [45]. This library "provides generic computational models of social behavior; such social behaviors include proxemics (social spacing), oculesics (eye gaze), kinesics (gesture), deixis (spatial referencing), prosodics (nonverbal speech cues), and speech." In the future, open-source libraries such as this may provide an easy means of integrating new functionality for implementations of the framework.

5.4 Limitations of the Project and Future Work

The current framework will accommodate a wide range of SAR designs; however, there are important limitations to the final system. Initial validation of SAR systems (particularly with younger or cognitively impaired participants) often involves Wizard-of-Oz pilot studies [46, 47, 48]. These studies mimic social agent/autonomous robot behavior by having an experimenter observe an interaction remotely (typically, from an adjacent room out of participant sight). The social agent behavior is guided by the experimenter and user responses are then incorporated into the final system design. Wizard-of-Oz studies ensure social agent behaviors are appropriate for and interpretable by the target population. The current framework is not designed for real-time operation by an experimenter. In order to facilitate such an interaction in a new implementation, the developer would need to include a new node or combination of nodes to interface with the experimenter separately, which would control feedback from the social agent directly.

Additionally, a range of SAR studies also incorporate multiple human users [49, 50, 51, 52]. These studies are typically used to evaluate joint attention, or to build social and communication skills for participants. The current design does not incorporate multiple human users; however, modification to do so would be relatively simple. The current User Interface node can be used as it is currently structured or duplicated to accept inputs from multiple users. In this case, only the Perception and Event Handler nodes would need to be updated to properly adjust performance based on the relevant human user(s).

Finally, full validation of the SAR system requires fault tolerancing. This process typically includes a pilot test of the system with a small sample ($n = 10$) of unimpaired, adult users. Fault tolerancing exposes any design flaws (e.g., code errors, problems

with data transmission, inter-node communication) that may adversely affect eventual use with the target population [53, 54]. Due to requirements for social-distancing and additional requirements with respect to institutional approval for human subjects research, it was not possible to find human subjects to test the two implementations of the framework for this project. Therefore, further testing of the system with unimpaired participants, and participants from the target population, must be completed once these regulations are lifted.

Future work on this project should focus on testing the system with a variety of users, developing the framework further by incorporating additional features, and adding more modular support for different types of hardware input and output.

BIBLIOGRAPHY

[1] "Average life expectancy in industrial and developing countries for those born in 2020, by gender," 2020. Accessed: 2020-5-26.

[2] S. Virani, A. Alonso, E. Benjamin, M. Bittencourt, C. Callaway, C. April, *et al.*, "Heart Disease and Stroke Statistics—2020 Update: A Report From the American Heart Association," *Circulation*, vol. 141, no. 9, pp. 139–596, 2020.

[3] J. Harris, "Geriatric Trends Facing Nursing with the Growing Aging," *Crit Care Nurs Clin North Am*, vol. 31, pp. 211–224, Jun 2019.

[4] C. Marras, J. Beck, *et al.*, "Prevalence of Parkinson's Disease across North America," *npj Parkinson's Disease*, vol. 4, no. 21, 2018.

[5] "Table A-10. Difficulties in physical functioning among adults aged 18 and over, by selected characteristics: United States, 2018," 2018. Accessed: 2020-2-01.

[6] A. Thomas, F. Al Zoubi, N. E. Mayo, S. Ahmed, F. Amari, A. Bussières, L. Letts, J. C. MacDermid, H. J. Polatajko, S. Rappolt, N. M. Salbach, M. F. Valois, and A. Rochette, "Individual and organizational factors associated with evidence-based practice among physical and occupational therapy recent graduates: A cross-sectional national study," *J Eval Clin Pract*, Dec 2020.

[7] I. Livingstone, J. Hefele, and N. Leland, "Physical and Occupational Therapy Staffing Patterns in Nursing Homes and Their Association with Long-stay

Resident Outcomes and Quality of Care," *J Aging Soc Policy*, pp. 1–19, Oct 2020.

[8] S. F. Atashzar, J. Carriere, and M. Tavakoli, "Review: How Can Intelligent Robots and Smart Mechatronic Modules Facilitate Remote Assessment, Assistance, and Rehabilitation for Isolated Adults With Neuro-Musculoskeletal Conditions?," *Front Robot AI*, vol. 8, p. 610529, 2021.

[9] M. E. Matsumoto, G. C. Wilske, and R. Tapia, "Innovative Approaches to Delivering Telehealth," *Phys Med Rehabil Clin N Am*, vol. 32, pp. 451–465, May 2021.

[10] T. M. Annaswamy, G. N. Pradhan, K. Chakka, N. Khargonkar, A. Borresen, and B. Prabhakaran, "Using Biometric Technology for Telehealth and Telerehabilitation," *Phys Med Rehabil Clin N Am*, vol. 32, pp. 437–449, May 2021.

[11] A. H. Chan, "Logistics of Rehabilitation Telehealth: Documentation, Reimbursement, and Health Insurance Portability and Accountability Act," *Phys Med Rehabil Clin N Am*, vol. 32, pp. 429–436, 05 2021.

[12] T. Jewell, "The Best Telemedicine Apps of 2020," 2020. Accessed: 2021-4-23.

[13] D. Feil-Seifer and M. Matarić, "Defining Socially Assistive Robotics," in *Proceedings of the IEEE 9th International Conference on Rehabilitation Robotics*, vol. 2005, pp. 465 – 468, 07 2005.

[14] J. Pirhonen, E. Tiilikainen, S. Pekkarinen, M. Lemivaara, and H. Melkas, "Can robots tackle late-life loneliness? Scanning of future opportunities and challenges in assisted living facilities," *Futures*, vol. 124, p. 102640, Dec 2020.

[15] N. Chen, J. Song, and B. Li, "Providing Aging Adults Social Robots' Companionship in Home-Based Elder Care," *J Healthc Eng*, vol. 2019, p. 2726837, 2019.

[16] F. O'Brolcháin, "Robots and people with dementia: Unintended consequences and moral hazard," *Nurs Ethics*, vol. 26, pp. 962–972, Jun 2019.

[17] Z. Zheng, J. Zhu, J. Fan, and N. Sarkar, "Design and System Validation of Rassle: A Novel Active Socially Assistive Robot for Elderly with Dementia," in *Proceedings of the 27th IEEE International Symposium on Robot and Human Interactive Communication*, pp. 1–4, 08 2018.

[18] J. Fasola and M. Matarić, "Using Socially Assistive Human–Robot Interaction to Motivate Physical Exercise for Older Adults," *Proceedings of the IEEE*, vol. 100, pp. 2512–2526, 08 2012.

[19] T. T. Lewis, H. Kim, A. Darcy-Mahoney, M. Waldron, W. H. Lee, and C. H. Park, "Robotic uses in pediatric care: A comprehensive review," *J Pediatr Nurs*, vol. 58, pp. 65–75, Dec 2020.

[20] A. Miguel Cruz, A. M. Ríos Rincón, W. R. Rodríguez Dueñas, D. A. Quiroga Torres, and A. F. Bohórquez-Heredia, "What does the literature say about using robots on children with disabilities?," *Disabil Rehabil Assist Technol*, vol. 12, pp. 429–440, 07 2017.

[21] N. Fitter, R. Funke, J. C. Pulido, L. Eisenman, W. Deng, M. Rosales, N. Bradley, B. Sargent, B. Smith, and M. Mataric, "Socially Assistive Infant-Robot Interaction: Using Robots to Encourage Infant Leg-Motion Training," *IEEE Robotics & Automation Magazine*, vol. PP, pp. 1–1, 04 2019.

[22] R. Krithiga, "Socially Assistive Robot for children with Autism Spectrum Disorder," in *2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE)*, pp. 1–4, 2019.

[23] S. Hajjaj and K. Sahari, "Developing portable HRI framework for open-source outdoor robots, through Selective Compartmentalization," 06 2019.

[24] Y. Mohamed and S. Lemaignan, "ROS for Human-Robot Interaction," 12 2020.

[25] C. L. Pollock, L. A. Boyd, M. A. Hunt, and S. J. Garland, "Use of the Challenge Point Framework to Guide Motor Learning of Stepping Reactions for Improved Balance Control in People With Stroke: A Case Series," *Physical Therapy*, vol. 94, pp. 562–570, 04 2014.

[26] P. Silva, F. Antunes, P. Graef, F. Cechetti, and A. Pagnussat, "Strength training associated with task-oriented training to enhance upper-limb motor function in elderly patients with mild impairment after stroke a randomized controlled trial," *American journal of physical medicine & rehabilitation / Association of Academic Physiatrists*, vol. 94, 08 2014.

[27] J.-H. Moon, J.-H. Jung, S.-C. Hahm, and H.-Y. Cho, "The effects of task-oriented training on hand dexterity and strength in children with spastic hemiplegic cerebral palsy: A preliminary study," *Journal of Physical Therapy Science*, vol. 29, pp. 1800–1802, 10 2017.

[28] M. Wälchli, J. Ruffieux, Y. Bourquin, M. Keller, and W. Taube, "Maximizing Performance: Augmented Feedback, Focus of Attention, and/or Reward?," *Medicine and science in sports and exercise*, vol. 48, 11 2015.

[29] A. Nagata, K. Doma, D. Yamashita, H. Hasegawa, and S. Mori, "The effect of augmented feedback type and frequency on velocity-based training-induced

adaptation and retention," *Journal of Strength and Conditioning Research*, vol. 34, p. 1, 02 2018.

[30] C. M. Cirstea, A. Ptito, and M. F. Levin, "Feedback and cognition in arm motor skill reacquisition after stroke," *Stroke*, vol. 37, pp. 1237–1242, May 2006.

[31] C. Breazeal and C. Kidd, "Designing for long-term human-robot interaction and application to weight loss," 2008.

[32] J. Wainer, D. J. Feil-seifer, D. A. Shell, and M. J. Mataric, "The role of physical embodiment in human-robot interaction," in *ROMAN 2006 - The 15th IEEE International Symposium on Robot and Human Interactive Communication*, pp. 117–122, 2006.

[33] F. Michaud, Y. Brosseau, C. Côté, D. Létourneau, P. Moisan, A. Ponchon, C. Raïevsky, J.-M. Valin, E. Beaudry, and F. Kabanza, "Modularity and integration in the design of a socially interactive robot," 09 2005.

[34] V. Rajendran, P. Carreno-Medrano, W. Fisher, A. Werner, and D. Kulić, "A Framework for Human-Robot Interaction User Studies," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6215–6222, 2020.

[35] P. Yeap Loh, W. Liang Yeoh, H. Nakashima, and S. Muraki, "Impact of keyboard typing on the morphological changes of the median nerve," *J Occup Health*, vol. 59, pp. 408–417, Sep 2017.

[36] H. J. Chong, S. J. Kim, and G. E. Yoo, "Differential effects of type of keyboard playing task and tempo on surface EMG amplitudes of forearm muscles," *Front Psychol*, vol. 6, p. 1277, 2015.

[37] M. A. Statton, M. Encarnacion, P. Celnik, and A. J. Bastian, "A Single Bout of Moderate Aerobic Exercise Improves Motor Skill Acquisition," *PLoS One*, vol. 10, no. 10, p. e0141393, 2015.

[38] H. Stranda, M. Haga, H. Sigmundsson, and H. Lorås, "The Effect of Aerobic Exercise on Speed and Accuracy Task Components in Motor Learning," *Sports (Basel)*, vol. 7, Feb 2019.

[39] A. J. Chen and F. Loya, "Strengthening goal-directed functioning after traumatic brain injury," *Handb Clin Neurol*, vol. 163, pp. 435–456, 2019.

[40] D. Elliott and S. J. Bennett, "Intermittent Vision and Goal-Directed Movement: A Review," *J Mot Behav*, vol. 53, no. 4, pp. 523–543, 2021.

[41] D. Elliott, J. Lyons, S. J. Hayes, J. J. Burkitt, S. Hansen, L. E. M. Grierson, N. C. Foster, J. W. Roberts, and S. J. Bennett, "The multiple process model of goal-directed aiming/reaching: insights on limb control from various special populations," *Exp Brain Res*, vol. 238, pp. 2685–2699, Dec 2020.

[42] M. Alt Murphy, M. C. Baniña, and M. F. Levin, "Perceptuo-motor planning during functional reaching after stroke," *Exp Brain Res*, vol. 235, pp. 3295–3306, 11 2017.

[43] C. L. Yang, R. A. Creath, L. Magder, M. W. Rogers, and S. McCombe Waller, "Impaired posture, movement preparation, and execution during both paretic and nonparetic reaching following stroke," *J Neurophysiol*, vol. 121, pp. 1465–1477, 04 2019.

[44] J. C. Stewart, R. Lewthwaite, J. Rocktashel, and C. J. Winstein, "Self-efficacy and Reach Performance in Individuals With Mild Motor Impairment Due to Stroke," *Neurorehabil Neural Repair*, vol. 33, pp. 319–328, 04 2019.

[45] E. T. Kaszubski, "The social behavior library," 2012.

[46] C. Sirithunge, A. G. B. P. Jayasekara, and D. P. Chandima, "An Evaluation of Human Conversational Preferences in Social Human-Robot Interaction," *Appl Bionics Biomech*, vol. 2021, p. 3648479, 2021.

[47] E. A. Björling, K. Thomas, E. J. Rose, and M. Cakmak, "Exploring Teens as Robot Operators, Users and Witnesses in the Wild," *Front Robot AI*, vol. 7, p. 5, 2020.

[48] G. Lakatos, M. Gácsi, V. Konok, I. Brúder, B. Bereczky, P. Korondi, and A. Miklósi, "Emotion attribution to a non-humanoid robot in different social situations," *PLoS One*, vol. 9, no. 12, p. e114207, 2014.

[49] K. Kompatsiari, F. Bossi, and A. Wykowska, "Eye contact during joint attention with a humanoid robot modulates oscillatory brain activity," *Soc Cogn Affect Neurosci*, vol. 16, pp. 383–392, Mar 2021.

[50] C. Willemse and A. Wykowska, "In natural interaction with embodied robots, we prefer it when they follow our gaze: a gaze-contingent mobile eyetracking study," *Philos Trans R Soc Lond B Biol Sci*, vol. 374, p. 20180036, 04 2019.

[51] P. Vogt, M. de Haas, C. de Jong, P. Baxter, and E. Krahmer, "Child-Robot Interactions for Second Language Tutoring to Preschool Children," *Front Hum Neurosci*, vol. 11, p. 73, 2017.

[52] S. Ivaldi, S. M. Anzalone, W. Rousseau, O. Sigaud, and M. Chetouani, "Robot initiative in a team learning task increases the rhythm of interaction but not the perceived engagement," *Front Neurorobot*, vol. 8, p. 5, 2014.

[53] A. Langer, R. Feingold-Polak, O. Mueller, P. Kellmeyer, and S. Levy-Tzedek, "Trust in socially assistive robots: Considerations for use in rehabilitation," *Neuroscience & Biobehavioral Reviews*, vol. 104, pp. 231–239, 2019.

[54] A. Tapus, M. J. Mataric, and B. Scassellati, "Socially assistive robotics [grand challenges of robotics]," *IEEE Robotics Automation Magazine*, vol. 14, no. 1, pp. 35–42, 2007.